UNITED STATES PATENT AND TRADEMARK OFFICE

Commissioner for Patents
United States Patent and Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450
www.uspto.gov

BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

Application Number: 09/997,404
Filing Date: November 29, 2001
Appellant(s): SREENIVASAN ET AL.

Rodney Lacy
For Appellant

EXAMINER'S ANSWER

This is in response to the appeal brief filed March 25, 2008 appealing from the Office action mailed May 3, 2007.

(1) Real Party in Interest

A statement identifying by name the real party in interest is contained in the brief.

(2) Related Appeals and Interferences

The examiner is not aware of any related appeals, interferences, or judicial proceedings which will directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

(3) Status of Claims

The statement of the status of claims contained in the brief is correct.

(4) Status of Amendments After Final

No amendment after final has been filed.

(5) Summary of Claimed Subject Matter

The summary of claimed subject matter contained in the brief is correct.

(6) Grounds of Rejection to be Reviewed on Appeal

The appellant's statement of the grounds of rejection to be reviewed on appeal is correct.

(7) Claims Appendix

The copy of the appealed claims contained in the Appendix to the brief is correct.

(8) Evidence Relied Upon

US 6189111 B1	Alexander; James R. et al.	2-2001
US 6145089 A	Le; Hung et al.	11-2000
US 6438705 B1	Chao; Ching-Yun et al.	8-2002

Microsoft Computer Dictionary, Fifth Edition; Microsoft Press; Copyright 2002; selected pages.

Official Notice has been taken that the use of a shell script or a Perl script as detailed in claims 11 and 12 are well known and expected in the art. Appellant has failed to seasonably traverse this assertion and therefore has been taken as Applicant's Admitted Prior Art for the duration of prosecution.

Official Notice is taken that a load-balancing event in the absence of a node failure as detailed in claims 15 and 17 is well known and expected in the art. Appellant has failed to seasonably traverse this assertion and therefore is taken as Applicant's Admitted Prior Art for the duration of prosecution.

(9) Grounds of Rejection

The following ground(s) of rejection are applicable to the appealed claims:

1. Claims 1, 2, 4, 5, 7-12, and 14-19 are rejected under 35 U.S.C. 103(a) as being unpatentable over Alexander et al. (US 6,189,111) (hereinafter Alexander) in view of Le et al. (US 6,145,089) (hereinafter Le) in view of Chao et al. (USPN 6,438,705) (hereinafter Chao).

Alexander taught a method and system to enhance survivability of system software components, even in the event of catastrophic failure of the computing element on which they reside. See abstract.

Regarding claim **1**, Alexander taught a system for implementing a failover policy comprising: a cluster infrastructure for managing a plurality of nodes; a high availability infrastructure for providing group and cluster membership services (cluster membership service or CLMS) **(column 5 lines 35-40)**; and a high availability script execution component (i.e. routine execution mechanism) (col. 6, lines 40-50) operative to execute computer code and receiving at least one failover attribute (failing node information and harvested data)

Alexander teachings are further operative to produce a runtime failover domain from an initial failover domain (recognizing the failing node and removing it from the bitmap) **(column 5 lines 40-50, column 6 lines 19-21, column 8 lines 40-42, column 9 lines 32-33)**. Note that Alexander teaching describe producing a failover domain when a failing node is recognized and a notification is send to the other nodes which represents a failover domain that by definition is the area of control to which the system will automatically transfer activity to a standby server upon failure of an active server.

Alexander did not specifically state that upon the detection of a failover event, executing a failover script comprising a set of one or more commands. In analogous art, Le discloses another system for implementing a failover policy which, upon the detection of a failover event, executing a failover script which includes a response to a service failure (col. 4, lines 37-60; col. 5, lines 53-67; Figure 4, ref. 450). It would have been obvious to one of ordinary skill in the art to combine the teaching of Le with Alexander in order to provide a high availability service process which provides services without requiring duplication of services as supported by Le (col. 1, lines 27-30).

Alexander in view of Le did not specifically disclose executing one or more action scripts in order to cause the resources to failover to the runtime domain. In analogous art, Chao discloses executing action scripts (i.e. node_down event) which causes the resources to failover to the runtime domain (i.e. the resources are restarted on the next_node, which is determined based on the preferred node to run the resource group based on node status, group preferred node list, and failover policy) (Figures 4, 4a, 4b; col. 16, lines 7-50). It would have been obvious to one of ordinary skill in the art to combine the teaching of Chao with Alexander and Le in order to allow support of a failover from one node to another in a multicluster environment as supported by Chao (col. 5, liens 14-18).

Alexander modified by Le and Chao is hereinafter referenced to as 'the first combination'.

Regarding claim **2**, the first combination further taught a method for determining a target node for a failover, comprising: executing a failover script, said script producing a failover domain, said failover domain having an ordered list of nodes **(column 5 lines 40-47, fig. 3 and column 9 lines 40-53 [ordered list of nodes])**; receiving a failover attribute **(column 5 lines 54-65)**; and based on the failover attribute and failover domain selecting a node upon which to locate a resource **(column 5 lines 50-52, column 9 lines 26-39 and column 10 lines 48-57)**.

Regarding claims 3 and 13, it is understood that in the first combination (Alexander: column 5, lines 40-47) an initial domain exist and is represented by "each of the other nodes". Further, when a node fails to respond the rest of the nodes are notified (Alexander: column 5 lines 47-50) and when the failed node is removed from the map and as per the described action, it is understood that a failover domain is effectively produced (Alexander: column 6 lines 54-56, column 8 lines 40-42, column 9 lines 32-33).

Regarding claim 4, the first combination further taught defining a resource group (Alexander: column 6 lines 54-56); and associating the failover script and the failover attribute with the resource group (Alexander: column 6 lines 56-63).

Regarding claim 5, the first combination taught the use of table to determine a non-failed node capable of harvesting data from a failed node. It is well known in the art of computer networks that files or tables are read in sequence or by an index and using the first node found in the table would be the conventional way to select a non-failed node (Alexander: column 6, lines 54-63) from a table.

Regarding claim 6, the first combination taught that the table is used either by the failed node performing active harvesting or by the non-failed node performing passive harvest (Alexander: column 6, lines 60-63) , which is commensurate with executing an

action script (Cluster Manager component (Regroup) or Harvest processes) for a target node.

Regarding claim 7, the first combination further taught action scripts verifying resources and resources configuration on a node (Alexander: column 9 lines 18-25) in the form of ASSERT macros that perform consistency checks and can be used on candidates for harvesting; and can further be used by applications for validation purposes.

Regarding claims 8-10, the first combination taught that operating systems of the embodiments are capable of encountering errors or faults (Alexander: column 6, lines 5-7), and further taught macros that perform validations (Alexander: column 9 lines 18-25) as explained above in relation to claim 7, further services running in a distributed fashion are taught (Alexander: column 5, lines 35-38), in addition to other services (Alexander: column 10, lines 5-11). It is understood that the first combination taught or at least suggest, that applications can validate data structures (Alexander: column 9, lines 24-25) that are dependent on services/applications/resources (Alexander: column 10, lines 5-11) and that upon discovering a particular status an appropriate arbitrary action can be performed (Alexander: column 9, lines 18-23) and that such action can be a recovery attempt (Alexander: column 6, lines 5-7), which when refereeing to a running service/resource/application will be the arbitrary preferred command of stopping or starting such service/resource/application.

Regarding claims 11 and 12 Examiner takes Official Notice (see MPEP § 2144.03) that "the use of a shell script or a Perl script" in a computer-networking environment was well known in the art at the time the invention was made. The Applicant is entitled to traverse any/all official notice taken in this action according to MPEP § 2144.03. However, MPEP § 2144.03 further states "See also In re Boon, 439 F.2d 724, 169 USPQ 231 (CCPA 1971) (a challenge to the taking of judicial notice must contain adequate information or argument to create on its face a reasonable doubt regarding the circumstances justifying the judicial notice)." Specifically, In re Boon, 169 USPQ 231, 234 states "as we held in Ahlert, an applicant must be given the opportunity to challenge either the correctness of the fact asserted or the notoriety or repute of the reference cited in support of the assertion. We did not mean to imply by this statement that a bald challenge, with nothing more, would be all that was needed". Further note that 37 CFR § 1.671(c)(3) states "Judicial notice means official notice". Thus, a traversal by the Applicant that is merely "a bald challenge, with nothing more" will be given very little weight.

Referring to claim 14, Alexander discloses the failover event comprises failure of a node (col. 9, lines 30-35).

Referring to claim 16, Alexander discloses the failover event is a load-balancing event (the Office construes the term "load-balancing event" to be "any event which requires rebalancing of the incoming requests", by this rationale, a nodal failure would constitute a load-balancing event, since the other nodes would have to shoulder the burden for the failed node) (col. 9, liens 30-35).

Referring to claims 15 and 17, the combination describes the invention substantively as described in the claims above. The combination fails to disclose the failover event is a load-balancing event in the absence of a node failure, however load balancing is a well known feature in resource allocation. By this rationale, "Official Notice" is taken that both the concepts and advantages of providing load balancing are well known and expected in the art. It would have been obvious to one of ordinary skill in the art to modify the combination to incorporate load balancing in order to reduce congestion on overloaded servers, thereby providing a balanced load on the plurality of servers.

Referring to claim 18, the combination discloses the invention substantively as described in claim 3. Furthermore the combination discloses saving the run-time failover domain (i.e. the nodes which are not failed; "remove the failed node from the bitmap") (Alexander: col. 10, lines 48-57). Furthermore detecting a second failover event would be obvious to one of ordinary skill in the art since it would have been

obvious to repeat the method multiple times for multiple effects. See St. Regis Paper Co. v. Bemis Co., 193 USPQ 8 (7th Cir. 1977). Furthermore the bitmap is used as input to the failover script to determine which node is available for harvesting. By this rationale, on the second failover event, the bitmap (with the failed node removed) will be used as input to the routine which determines which other node (which has not failed) is available to harvest the objects of the failed node.

Referring to claim 19, the combination discloses the resource includes an application and comprising an application plug-in (i.e. an interface utilized to communicate with a user or client application) which provides a high-availability interface for the application (Alexander: col. 3, lines 8-20).

(10) Response to Argument

Appellant's arguments (Brief, pages 12-15) have been fully considered and are refuted below.

Appellant argues, in substance, that Alexander fails to disclose "a high availability script execution component" which interprets a failover script. The Examiner agrees. Alexander discloses a routine execution mechanism which executes a routine to investigate an error and take an action. The Examiner never stated that the routine execution component interprets a failover script, rather this step is provided by Le (see

rejections above). Alexander discloses an execution mechanism which is able to provide a platform for the failover script to execute upon in order to provide the failover domain. By this rationale, the rejection should be maintained.

Appellant argues, in substance, that Alexander teaches away from the use of scripts in view of the execution of binary code (e.g. a subroutine) rather than script execution. The Examiner disagrees. The Examiner contends that a routine can comprise a script. Microsoft Computer Dictionary defines a "routine" as "any section of code that can be invoked (executed) within a program". Nowhere in this definition is the explicit limitation of "binary code", and therefore Alexander does not teach away from script execution. By this rationale, the rejection should be maintained.

Appellant argues, in substance, that Alexander does not teach the removal of a node from the bitmap by a failover script, rather the rote removal of the node. The Examiner agrees. The Examiner has already described in previous Office Actions that Alexander does not teach the use of a failover script, rather is using the Le and Chao reference to refute that particular limitation. Applicant's attention is directed to Alexander: col. 10, lines 48-58 which clearly discloses that a failed node is removed from the bitmap and to Le: col. 4, lines 37-60; col. 5, lines 53-67 which discloses executing a failover script in response to a service failure. Appellant is reminded that one cannot show nonobviousness by attacking references individually where the rejections are based on combinations of references. See *In re Keller*, 642 F.2d 413,

208 USPQ 871 (CCPA 1981); *In re Merck & Co.*, 800 F.2d 1091, 231 USPQ 375 (Fed. Cir. 1986). It is the combination of Alexander in view of Le and Chao which disclose the claimed invention. Merely pointing out that Alexander does not disclose the whole of the claim amounts to nothing since the rejection is based on a number of references. By this rationale, the rejection should be maintained.

Appellant argues, in substance, that Le does not teach the use of a failover script, rather an action script as defined in the specification. The Examiner disagrees. As defined in the specification, a "failover script" is "a shell script which generates a run-time failover domain" (Specification, page 9). An "action script" as defined in the specification is "a script that determines how a resource is started, monitored, and stopped" which includes the scripts "start, stop, monitor, and restart" (Specification, pages 9-10). Applicant is misinterpreting the script file of Le, stating that it is an 'action script'. However this script includes a response to service failure and may *include scripts* to start, stop, and restart the service. Each of these commands (i.e. start, stop, and restart) are the *action scripts*. The only requirement of a failover script, as defined in the specification is that it is a shell script that generates a run-time failover domain. As evidenced by the rejection above. The script file 450 does exactly that (i.e. provide a response to service failure). It does not deal with the actual starting and restarting of services. That is left for the *action scripts* start, stop, and restart. The script file, calls these action scripts, and therefore can be reasonably construed as a "failover script". By this rationale, the rejection is maintained

Appellant argues, in substance, that the ASSERT command of Alexander does not verify that the resource is configured on the target node. The Examiner disagrees. Appellant's attention is directed to Col. 9, lines 24-25 of Alexander which discloses that "applications can use all the same techniques *to validate its own data structures*". This clearly teaches that the application is capable of verifying its own data structures, which would include the configuration on other devices. By this rationale, the rejection should be maintained.

Appellant argues, in substance, that Alexander, Le and Chao do not disclose an application plug-in that provides a high-availability interface for the application. The Examiner disagrees. Appellant has not defined what is meant by an "application plug-in" and therefore broad consideration has been taken. As such, the Examiner's interpretation as provided in the rejection is proper and therefore the rejection should be maintained.

(11) Related Proceeding(s) Appendix

No decision rendered by a court or the Board is identified by the examiner in the Related Appeals and Interferences section of this examiner's answer.

For the above reasons, it is believed that the rejections should be sustained.

Respectfully submitted,

/Joseph E. Avellino/

Primary Examiner, Art Unit 2146

Conferees:

/Jeffrey Pwu/

Supervisory Patent Examiner, Art Unit 2146

/John Follansbee/

Supervisory Patent Examiner, Art Unit 2151